ECEN 4638

Controls Lab 3

Austin Glaser

Jay Greco

1.2.1 → 1.2.3 System Identification Tasks

We completed the initial system identification tasks as part of lab 2. Please see our lab 2 in order to verify our methods. Our estimated system parameters are included below.

Variable	Estimated Value
J_1	0.1 $Kg\text{-}m^2$
J_2	0.1 $Kg\text{-}m^2$
J_3	0.1 $Kg\text{-}m^2$
c_1	0.015 $Kg\text{-}m^2\text{-}s^{-1}$
c_2	0.01 $Kg\text{-}m^2\text{-}s^{-1}$
c_3	0.01 $Kg\text{-}m^2\text{-}s^{-1}$
k_1	2.7 $N\text{-}m\text{-}rad^{-1}$ ($Kg\text{-}m^2\text{-}s^{-2}$)
k_2	2.7 $N\text{-}m\text{-}rad^{-1}$ ($Kg\text{-}m^2\text{-}s^{-2}$)

Table 1: Determined parameters from experiment 2

1.2.3 Hardware Gain

We determined the system's hardware gain both mathematically, and by adding it as a parameter in our gray-box identification, as suggested by the TA. The relationship between voltage and output torque isn't completely linear. The system will perform differently at different voltage levels: for example, close to zero, the output torque will be small as compared to the proportional voltage applied. Of course, the system saturates, and can only supply a finite amount of voltage and torque. We can bypass this issue by only operating our system in the close-to-linear region that we determined during the lab. Within our upper and lower limits, we can neglect the nonlinearities that are part of the system at low and high voltages.

1.2.5 System Identification Using MATLAB

We performed two system identification procedures as part of lab 3. The first was a "black box" estimation, where we only supplied our system with an input, observed the output, and estimated a transfer function and system parameters. For the black box system identification, we used various input signals, all constrained within the near-linear region of our system. These included: steps, square waves, sine waves, and sawtooth waves. See table 2 for all input signals used in the black box identification procedure. All were constrained to an amplitude between 0.15 to 0.3.

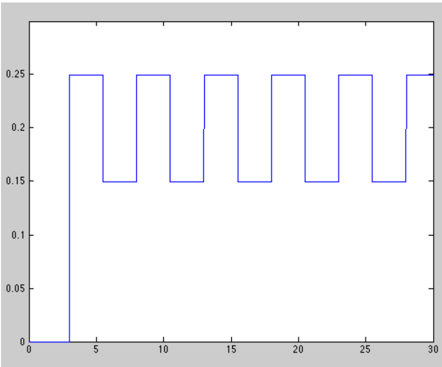
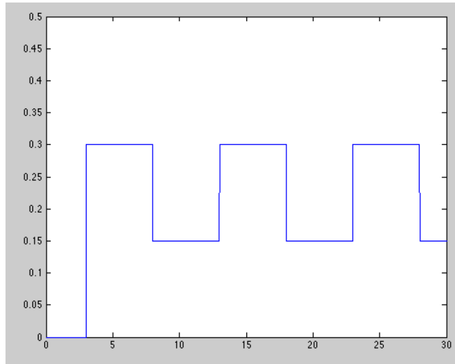
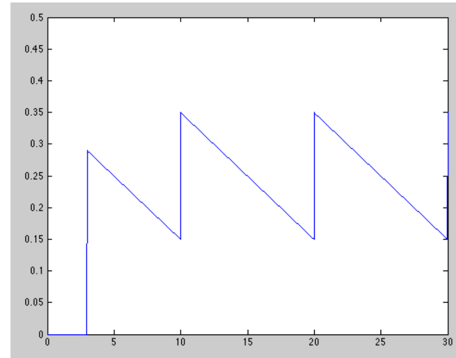
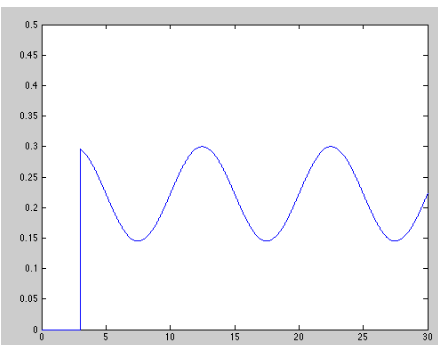
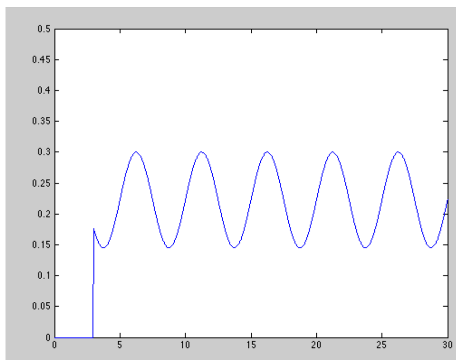
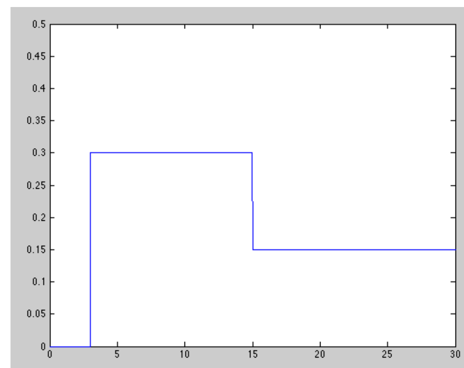
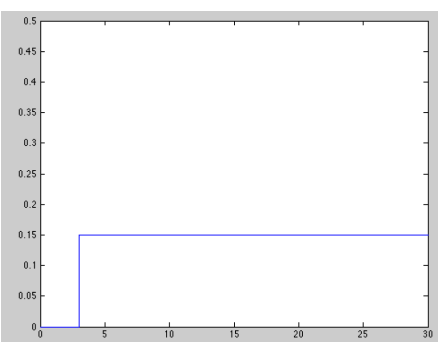
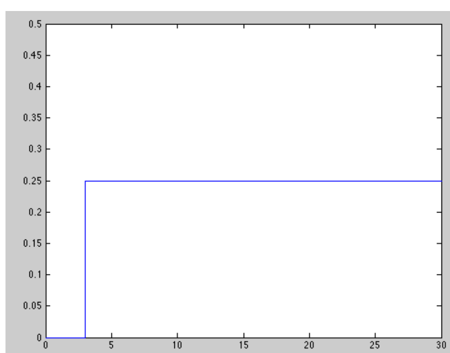
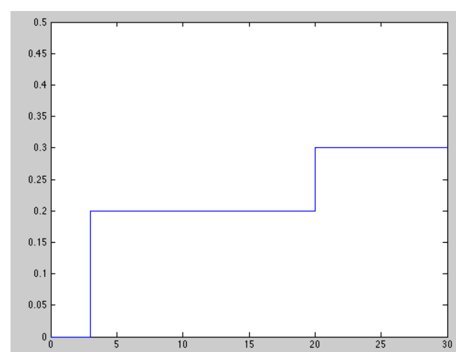
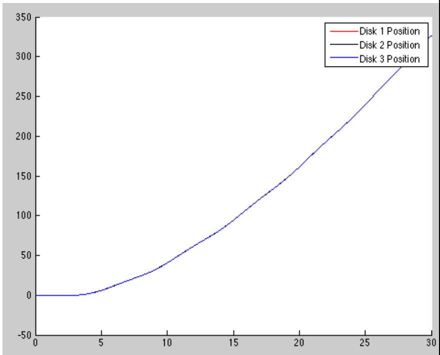
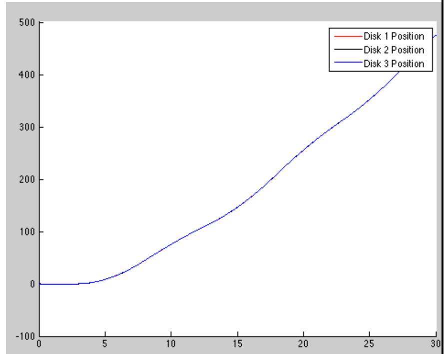
 Square wave, T = 5	 Square wave, T = 10	 Sawtooth, T = 10
 Sine wave, T = 10	 Sine wave, T = 5	 Up → Down step
 Simple step, amplitude = 0.15	 Simple step, amplitude = 0.25	 Up → Up step

Table 2: Selected system inputs for black box identification

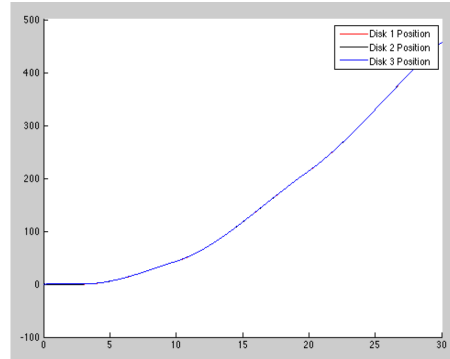
For each input, the response of all three disks was recorded. The response of all disks is shown in table 3.



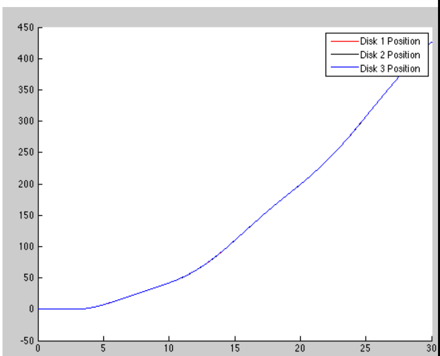
Square wave, T = 5, response



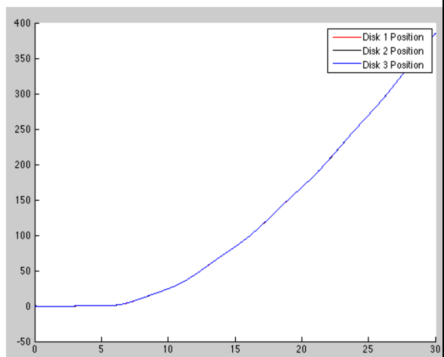
Square wave, T = 10, response



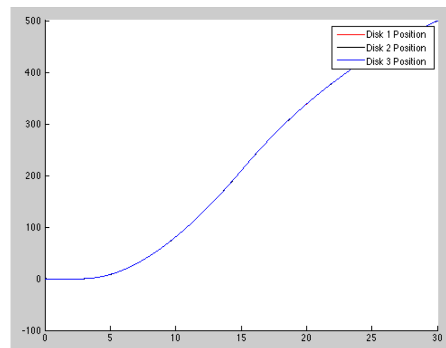
Sawtooth, T = 10, response



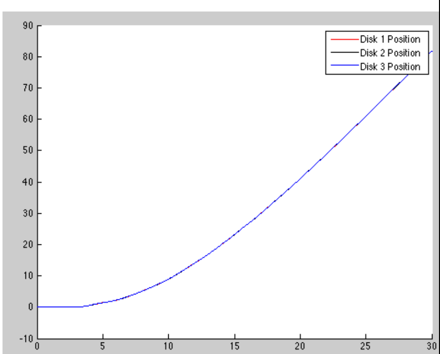
Sine wave, T = 10, response



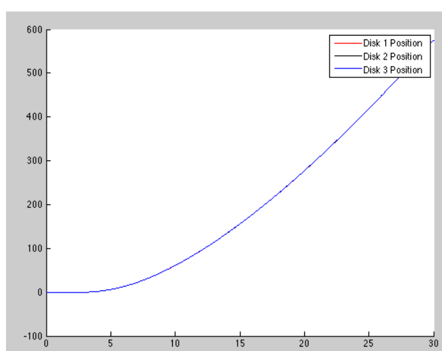
Sine wave, T = 5, response



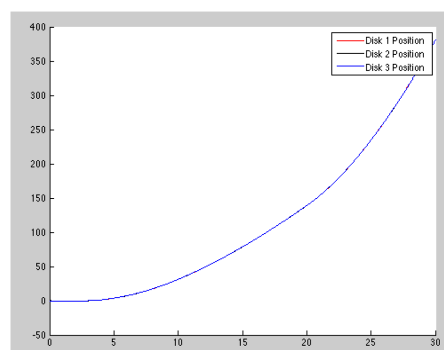
Up → Down step, response



Simple step, amplitude = 0.15



Simple step, amplitude = 0.25



Up → Up step, response

Table 3: System response to inputs given in table 2

In each case, the system's response was analyzed for a few things. Does the response appear to be taking place in the linear region? Is the system saturating or oscillating due to too large of an input? Does the given response make sense for the input given? For all responses, the responses physically agree with their inputs. For example: on the Up → Down step response, the system's position increases monotonically throughout the test, but the system velocity reflects the change in input signal; the velocity increases as the large initial step is applied, and then decreases as the step is reduced.

Next, the experimental data was passed into MATLAB. The entire dataset was combined into a data object, `data_combined.mat`, in order to perform a batch system identification. Next, data for each experiment and disk was split. Finally, data objects were created for each disk, and lastly, `tfest()` was called in order to perform the black box identification. Appendix 1 contains a sample of the black box identification. Figure 1 shows the output of the black box system identification procedure for experiment 2, the square wave input with T=10. The identification procedure generated a transfer function which matched the experimental data with a 99% fit for all three disks.

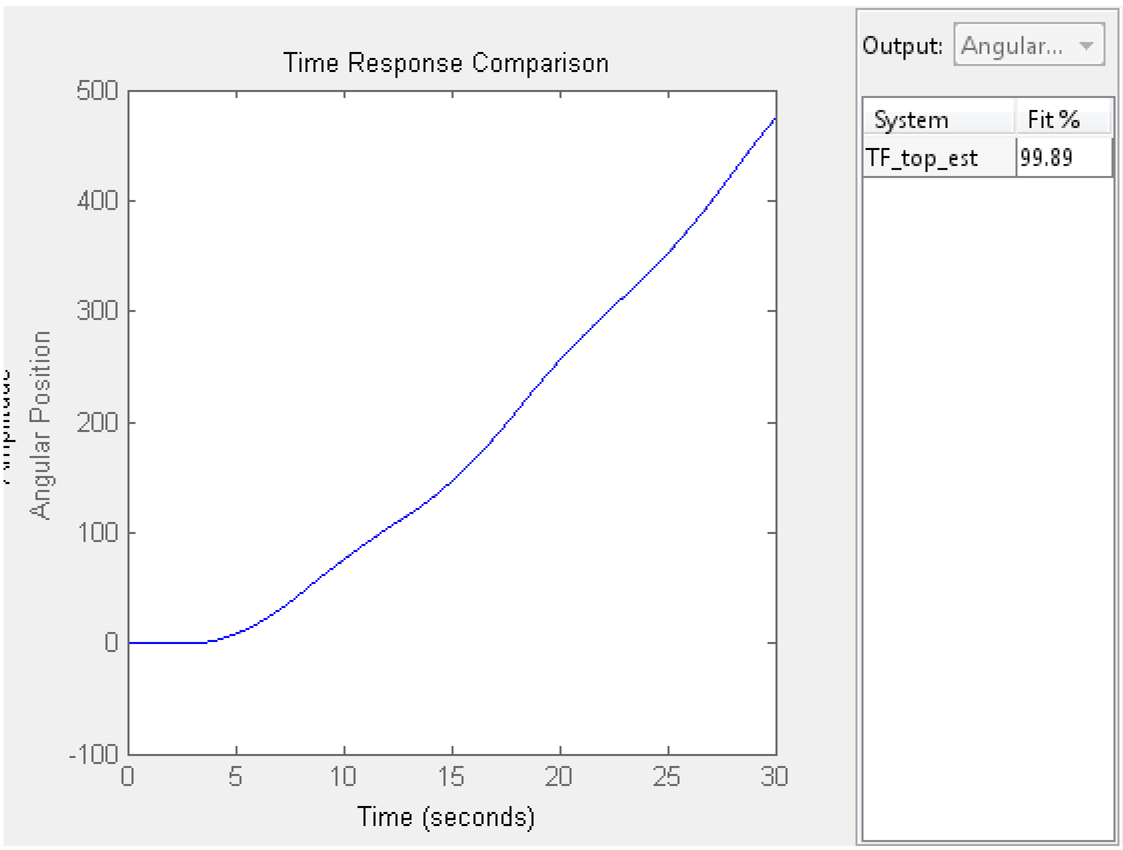


Figure 1: System identification results for a square wave input

It is worth noting that not all experiments produced accurate results. Several of the experiments, especially those operating at a higher frequency, did not generate transfer functions which accurately matched the experimental data. In addition, some of our inputs were hovering around the linear-nonlinear region boundary, which causes the system identification to fail. Figure 2 shows the result of a less-successful system identification.

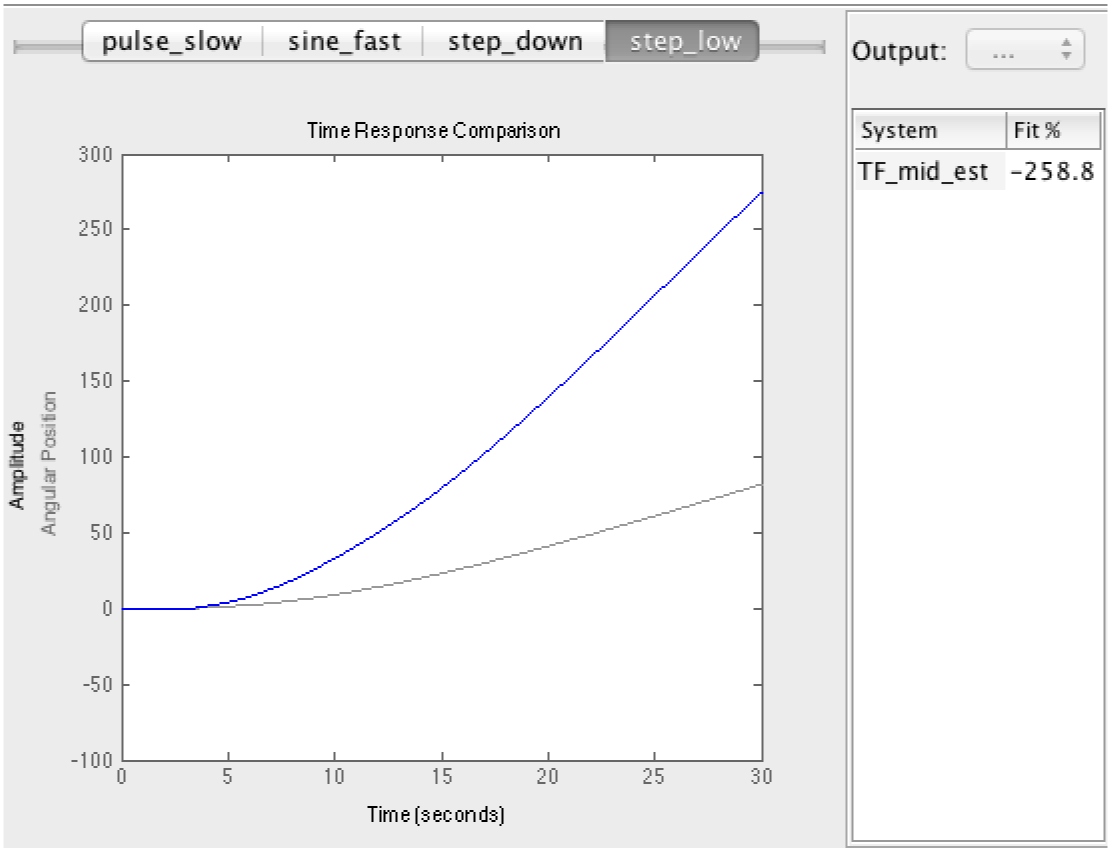


Figure 2: System identification results for a step input. The input was very close to the nonlinear region of the system, and therefore generated a poor estimated transfer function.

The second system identification procedure that was performed was a gray box estimation. The gray box system identification procedure uses partially-known system parameters in order to more successfully predict the parameters of the system. In this case, we have two options: use the system parameters estimated from lab 2, or, use the system parameters estimated from the lab 3 black box identification. First, the gray box identification was performed using the constants estimated in lab 2. As predicted in the end of lab 2, the parameters were close, but not very accurate. The gray box identification results are not a great fit to the experimentally measured data, as shown in figure 3.

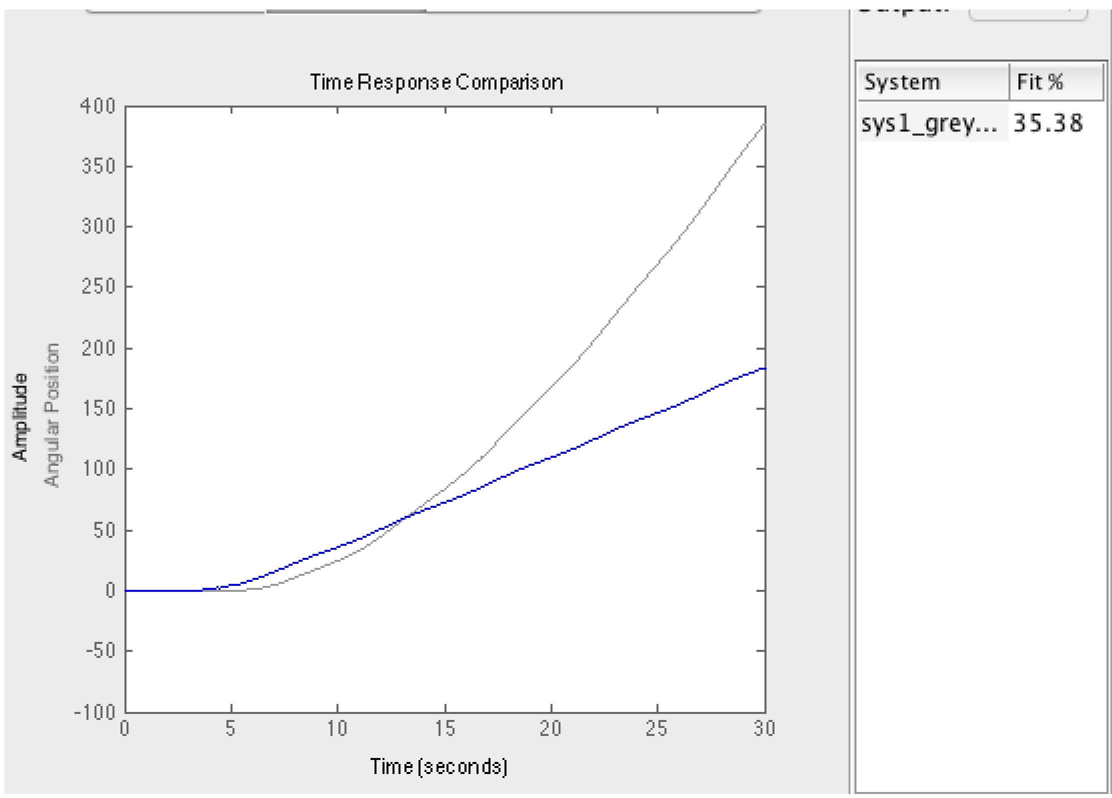


Figure 3: Gray box system identification results. The initial system parameters were not very accurate, and therefore resulted in a poor system identification.

It can be seen that while the estimation completes successfully, the estimated system is a poor fit to the physical system.

Rather than use the system parameters found in lab 2, the estimated system parameters from the black box identification can be used. In order to save the parameters calculated, the `getpvec()` MATLAB command is used. The parameters returned should, in theory, correspond with the parameters in the physical system. However, this relies on the assumption that the black box identification is successful enough to generate reasonable parameters. Furthermore, all input data must be valid, and if there is invalid data, it will throw off the generated parameters by a large amount. This appears to be the case for our system; even though the plots match with some degree of accuracy, they are not accurate enough to generate a full set of “correct” system parameters. After running our black box identification, the following parameter vector (left) is returned:

```
ans =

   10.9876
    9.5014
   55.2423   par =
   35.2569
   40.1226        2.7000
   12.3638        0.0200
    0.7456        0.0100
    4.9843        0.0200
    1.4540        2.7000
    2.1966        0.0100
    0.0056        0.0200
    0.0085        0.0100
    0.2300        1.0000
```

Comparing with the estimated parameter values (right), it's clear that the black box identification isn't accurate enough to generate reasonable parameter values. Some similarities can be seen, with the values of Kn residing between 1 and 5, Cn residing between 0.005 and 0.23, and one J near 0.7. The inaccuracy can be confirmed by running the gray box identification with these parameters rather than the lab 2 parameters. Figure 4 shows the result of the system identification with the black box estimated parameters:

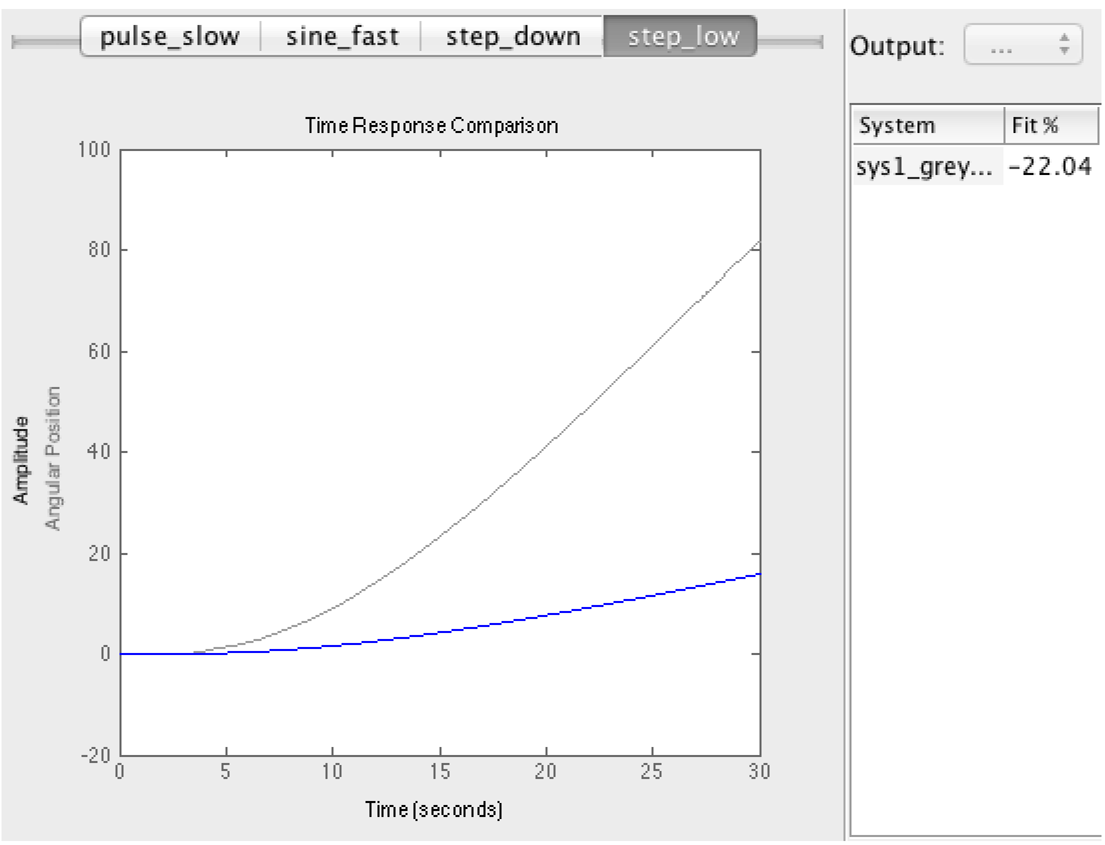


Figure 4: Gray box system identification results using the black box generated system parameters.

1.2.6 System Identification Analysis

As discussed, the black box generated parameters are just not close enough to use in an accurate gray box system estimation. Furthermore, it can be concluded that the estimated parameters from lab 2 are actually more accurate than the black box generated parameters, but not by much. All in all, it's obvious that more refinement is needed in the black box estimation in order to generate a usable gray box identification procedure. It is clear that the model is getting "better" (closer to the real system), but still needs improvement before it can be used to design a controller for this system. As for error approximation: the relative fit of the estimated transfer function to the measured response gives rise to information about how accurate the various parameters are. Better fit indicates more accurate parameters. Moreover, the physical effect each parameter has on the system can be used to determine where error lies in the system. If the system appears to have too much inertia, friction, or spring, it can indicate error within the K, C, and J constants.

Appendix 1: Sample Black Box System Identification Procedure

Contents

Clear out everything

```
clear all
close all
clc
```

Bring in Data

```
load('data_combined_new.mat')
```

Extract relevant experiments

```
top_experiments = [2];          % Which experiments to use for top
mid_experiments = [2];          % Which experiments to use for middle
bot_experiments = [2];          % Which experiments to use for bottom

name_comb_bot   = name_comb_bot(:,top_experiments);
name_comb_mid   = name_comb_mid(:,mid_experiments);
name_comb_top   = name_comb_top(:,bot_experiments);

y_comb_bot      = y_comb_bot(:,top_experiments);
y_comb_mid      = y_comb_mid(:,mid_experiments);
y_comb_top      = y_comb_top(:,bot_experiments);

u_comb_bot      = u_comb_bot(:,top_experiments);
u_comb_mid      = u_comb_mid(:,mid_experiments);
u_comb_top      = u_comb_top(:,bot_experiments);

intr_smpl_val_top = repmat({'zoh'}, size(name_comb_top));
intr_smpl_val_mid = repmat({'zoh'}, size(name_comb_mid));
intr_smpl_val_bot = repmat({'zoh'}, size(name_comb_bot));
```

Create Objects

```
u_t_id_top = iddata(y_comb_top,u_comb_bot,[],'ExperimentName',name_comb_bot,'Domain','Time','S
amplingInstants',t,'InterSample',intr_smpl_val_top,'TimeUnit','seconds','InputName',{'Torque'}
,'InputUnit',{'Nm'},'OutputName',{'Angular Position'},'OutputUnit',{'Radians'});
```

```
u_t_id_mid = iddata(y_comb_mid,u_comb_mid,[],'ExperimentName',name_comb_mid,'Domain','Time','S
amplingInstants',t,'InterSample',intr_smpl_val_mid,'TimeUnit','seconds','InputName',{'Torque'}
,'InputUnit',{'Nm'},'OutputName',{'Angular Position'},'OutputUnit',{'Radians'});
u_t_id_bot = iddata(y_comb_bot,u_comb_bot,[],'ExperimentName',name_comb_top,'Domain','Time','S
amplingInstants',t,'InterSample',intr_smpl_val_bot,'TimeUnit','seconds','InputName',{'Torque'}
,'InputUnit',{'Nm'},'OutputName',{'Angular Position'},'OutputUnit',{'Radians'});
opt = compareOptions('InitialCondition','zero');
```

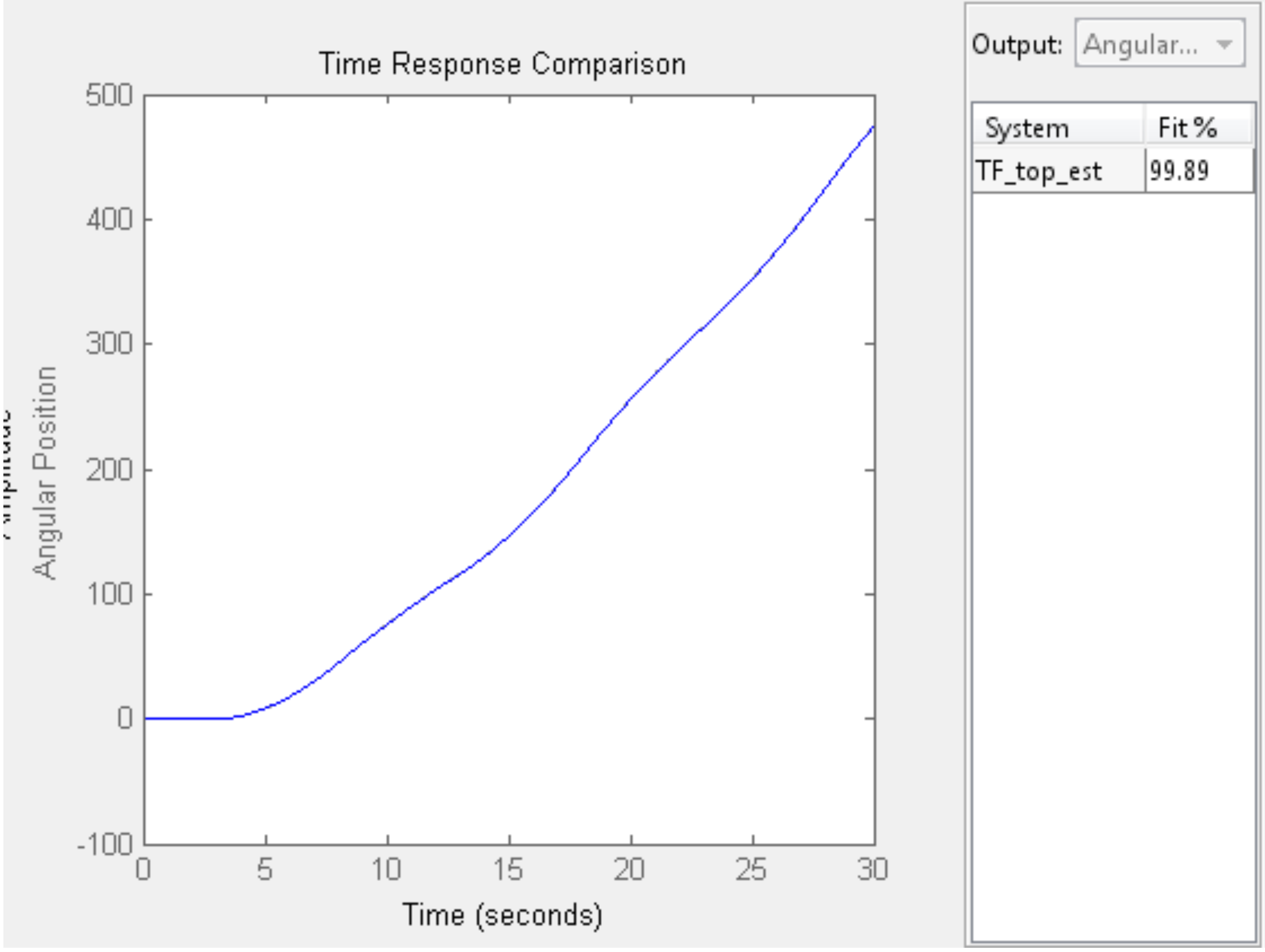
Black Box Identification

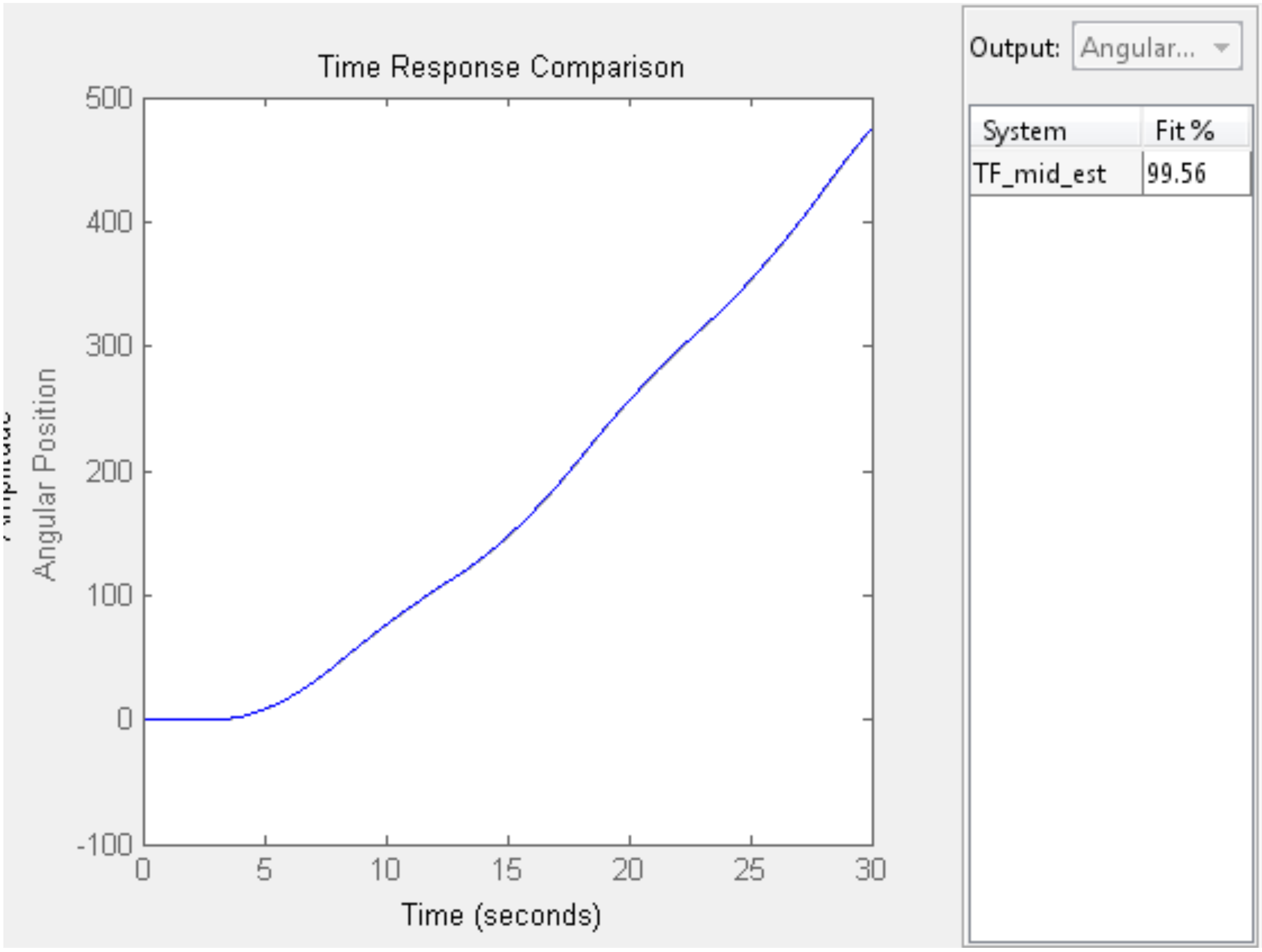
```
iodelay = NaN;
TF_top_est=tfest(u_t_id_top,6,5,iodelay,'Feedthrough','false');
TF_mid_est=tfest(u_t_id_mid,6,5,iodelay,'Feedthrough','false');
TF_bot_est=tfest(u_t_id_bot,6,5,iodelay,'Feedthrough','false');
```

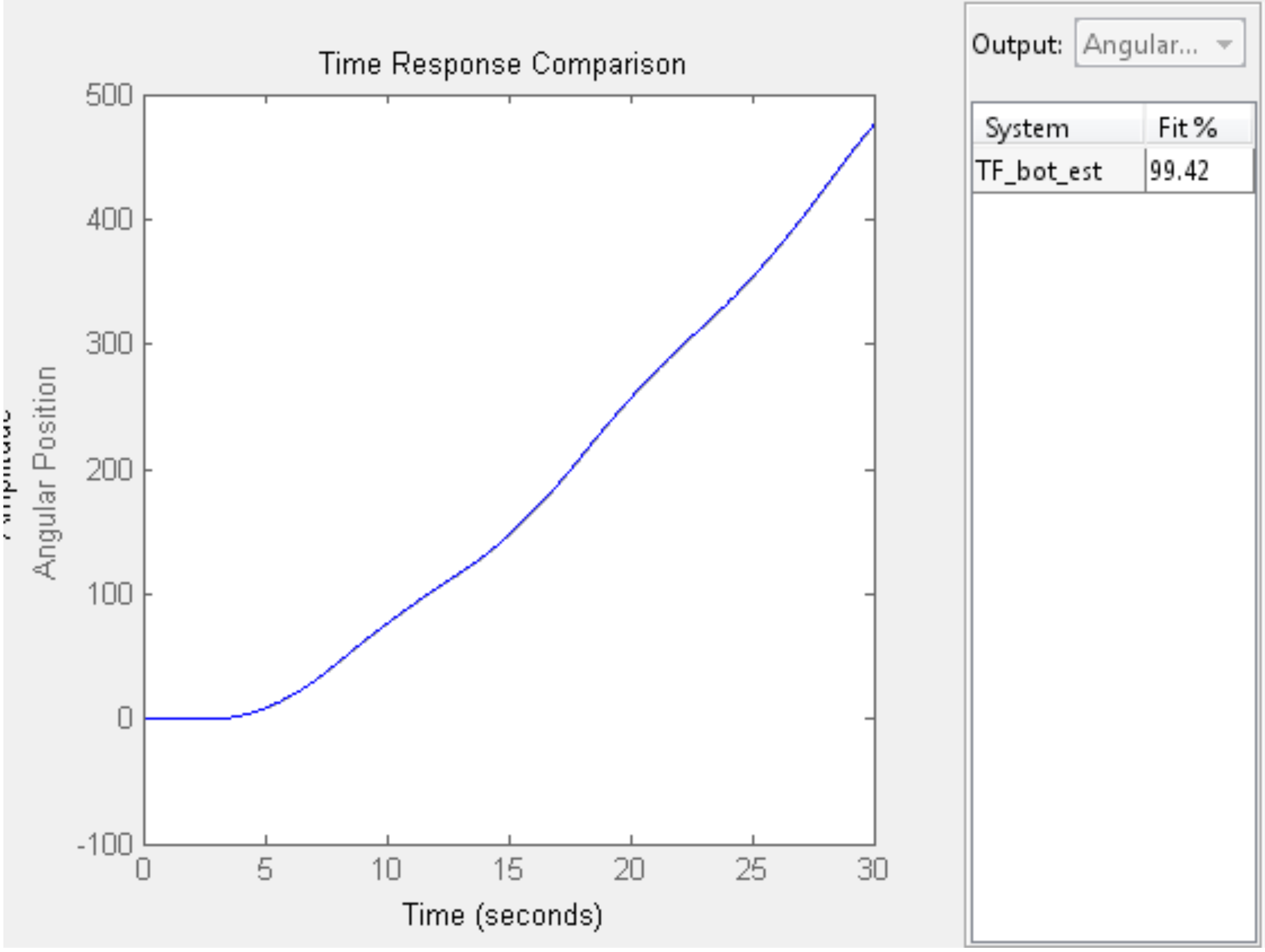
```
Warning: The "feedthrough" option is ignored for continuous-time estimation. Use
NZ<NP to enforce absence of feedthrough.
Warning: The "feedthrough" option is ignored for continuous-time estimation. Use
NZ<NP to enforce absence of feedthrough.
Warning: The "feedthrough" option is ignored for continuous-time estimation. Use
NZ<NP to enforce absence of feedthrough.
```

Comparison for Black Box

```
figure(1)
compare(u_t_id_top,TF_top_est,Inf,opt)
figure(2)
compare(u_t_id_mid,TF_mid_est,Inf,opt)
figure(3)
compare(u_t_id_bot,TF_bot_est,Inf,opt)
%getpvec(TF1_est)
%advice(uin_tin_id);
```


Time Response Comparison
Angular Position
Time (seconds)
Output: Angular...
System
Fit %
TF_top_est
99.89


Time Response Comparison
Angular Position
Time (seconds)
Output: Angular...
System
Fit %
TF_mid_est
99.56



Grey Box Identification

```
%{
K1=par(1,1);
J1=par(2,1);
C1=par(3,1);
J2=par(4,1);
K2=par(5,1);
C2=par(6,1);
J3=par(7,1);
C3=par(8,1);
kh=par(9,1);
%}
% par = [2.7;0.02;0.01;0.02;2.7;0.01;0.02;0.01;1];
% aux = {};
% T = 0;
% sys1_grey_id = idgrey('TDS',par,'c',aux,T);
% sys1_grey_est = greyest(u_t_id_top,sys1_grey_id);
```

Comparison for Grey Box

figure(4) compare(u_t_id_top,sys1_grey_est,Inf,opt) getpvec(sys1_grey_est)

Published with MATLAB® R2014a